AUTOMATIC SPEAKER VERIFICATION AND DIARIZATION ON VOXCELEB DATA COLLECTION

A Dissertation
Presented to
The Academic Faculty

By

Yufeng Yang

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in the
School of Electrical and Computer Engineering

Georgia Institute of Technology

May 2020

AUTOMATIC SPEAKER VERIFICATION AND DIARIZATION ON VOXCELEB DATA COLLECTION

Approved by:

Dr. David Anderson, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Mark Davenport
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Matthieu Bloch
School of Electrical and Computer Engineering
Georgia Institute of Technology

Date Approved: April 20th, 2020

To my great parents and those who always stay with me in my life journey.

ACKNOWLEDGEMENTS

I would like to thank my thesis advisor, Dr. David Anderson, who gave me the opportunity to conduct research and developed my interest in speech research through his guidance on my master thesis research. I want to thank my friend Desmond Caulley for his help introducing me to the speaker recognition and his collaboration on the project. I want to thank the professors and friends at Georgia Tech who have helped and taught me a lot during the past two years. Finally, I want to thank my dear parents, Mr. Xuewen Yang and Mrs. Qinhong Du, without whose love and support I could not reach my achievements today.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Automatic speaker verification (ASV) is increasingly getting more attention in speech research field in recent years. Because of the importance of cyber-security and personal property security, ASV can be used in many fields in the future in addition to fingerprint and face information. In ASV research, a variety of datasets are needed to train good models. Current datasets include NIST SRE, VoxCeleb, etc. In this work, to collect a non-English speaking dataset, the pipeline of VoxCeleb data collection is adopted to collect an East Asian language-speaking Celebrities (EACeleb) dataset. To remove some noisy segments of the output and make the dataset cleaner, speaker diarization is used in this research and the collected data is filtered. Due to the lack of ground truth labels of the collected data, ASV is used to measure the data cleanness improvement of our dataset. Equal error rate (EER) can be lowered by 25.63% after speaker diarization compared to the original EACeleb using a pretrained x-vector model for measurement. Also, by training the speaker verification using EACeleb data, when testing the EER performance, EACeleb after diarization can outperform VoxCeleb by 36.78%.

CHAPTER 1
INTRODUCTION AND BACKGROUND

With the development of artificial intelligence and machine learning, more and more models are published to help perform multiple tasks such as image classification, image captioning, facial recognition, and facial verification, etc. People make use of such technology in our daily life, for example, facial recognition or fingerprint recognition in smartphones, such as the iPhone. At the same time, speech signals are also getting more and more attention. Voice assistant is a useful tool for people to control cell phones or home appliances without touching them. For example, famous voice assistants such as Apple Siri, Amazon Alexa, Microsoft Cortana, etc. are commonly used. However, user-security is also a major concern for the use of speech signals. For example, a banking session may require a user to say a specific sentence to verify the transaction, Siri may recognize the user by hearing "Hey Siri." Thus, speaker recognition is being more and more important along with the speech signal.

Automatic speaker recognition is the identification of a person using voice characteristics. This is a task to answer "Who is speaking?" [1]. Usually, speaker recognition can be classified as speaker identification or speaker verification, where the difference is that, identification is to determine who is the speaker while verification is to tell whether the speaker is the same as the claimed speaker [2]. In this research speaker verification is the task to measure the performance.

Several contests have been held for speaker recognition such as the NIST speaker recognition evaluation (SRE), VoxCeleb, the speakers in the wild (SITW) speaker recognition challenge, etc [3, 4, 5] to build better speaker identification/verification models. With the development of deep neural network (DNN)-based speaker embedding, models can be trained to perform very well on specific datasets. New datasets are also needed for academia

and industry to improve current identification/verification approaches. Furthermore, the language of most datasets is English. Thus in this work, non-English speaking speakers are the focus and with the help of pipeline of VoxCeleb, a dataset named East Asian language-speaking Celebrities (EACeleb) is collected with speakers from China, Japan, and South Korea.

Automatic speaker verification and a brief history is introduced in Section 1.1. Then, speaker diarization is introduced in Section 1.3, which is used in our proposed method. Finally Section 1.2 introduces the process of VoxCeleb data collection procedure to help explain the basis of this thesis.

1.1 Automatic Speaker Verification

1.1.1 System Architecture

Automatic speaker verification (ASV) can be explained by the pipeline in Fig. 1.1 [6]. In the pipeline, representative feature parameters are extracted from the audio recording that aims to get the characteristics of the speaker. Features obtained from enrollment are usually used to build verification models that can help discriminate the user from other speakers. For test speakers, the corresponding features will be extracted from the pretrained system and then compared with the enrolled speakers. By computing a score between an enrolled speaker and a test speaker, we can decide to accept or reject the claimed identity. If the score is higher than some threshold that we defined, we accept the speaker. Otherwise, we reject the test speaker. Different from ASV, for speaker identification, the testing speaker is compared with multiple feature models to determine the best match while verification systems compare an utterance against a single voiceprint.

Usually, speaker verification has two categories: text-dependent and text-independent. In text-dependent verification, the text must be the same for enrollment and verification. Prompts can either be common across all speakers or unique. Text-independent verification is more commonly used as it requires very little, if any, cooperation by the speaker. In this

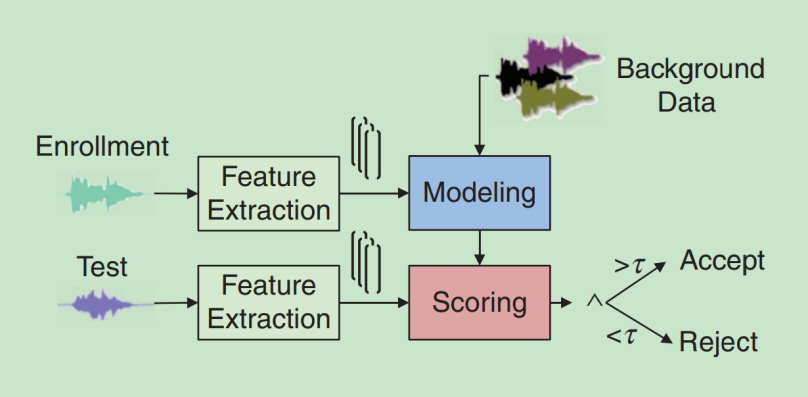

Figure 1.1: Basic speaker verification system.

case the text during enrollment and test can be different. In fact, the enrollment may happen without the user's knowledge, as is the case for many forensic applications.

1.1.2 Feature Parameter of ASV

In Section 1.1.1, features are extracted from the audio recording. Usually features are extracted as a fix-dimensional vector. Due to the variation of the length of utterances, some methods may be used to normalize the utterance. Different from image processing or signal processing in telecommunications, features of speech signals are usually extracted in a short-term manner. The most popular features for speech recognition and speaker recognition are Mel-frequency cepstral coefficients (MFCCs) [7] and linear predictive coefficients (LPCs) [8]. Also, with the need for speech security, some other features can also be considered in different schemes, such as inverted MFCCs (IMFCCs), and constant-Q cepstral coefficients (CQCCs) for spoofing detection [9]. In this thesis, MFCCs are the main focus.

Steps for extracting MFCC are as follows. First, the original utterance or audio samples are divided into short overlapping segments, usually 20-25 milliseconds. The signal obtained from each frame is then multiplied by a window function such as a Hanning or Hamming window, and the Fourier power spectrum is computed. After this, we take the logarithm of the Fourier power spectrum and use the Mel-space filter bank, which is nonlinearly spaced in the frequency domain to focus more on the low-frequency part. Focusing

more on the low-frequency part means the channels are more sensitive in the lower end of the spectrum, which is similar to the human auditory system. The logarithm can expand the scale of coefficients and also transform multiplications into additions. The filter bank produces the spectrum energy in each frequency band, also known as channels. Finally, the discrete cosine transform (DCT) is performed on the filter bank energies and we keep a specific number of parameters. The DCT is advantageous in the task because it can transform the energy of the signal into coefficients. Secondly, DCT can decorrelate the energies of the energy from short-term speech signal. We can improve the efficiency by only keeping part of the DCT coefficients while still keeping most of the features of the MFCCs. Generally, in the application of speaker verification, MFCCs are not enough. So the velocity and acceleration across multiple frames are appended to the MFCCs, which are also known as *deltas* and *delta-deltas*.

1.1.3 Front-End Approaches of ASV

In early research, vector quantization (VQ) was used [10] for speaker verification. Then Gaussian mixture models (GMMs) were proposed for speaker modeling [11]. A GMM is a combination of clusters of probability density functions (PDFs). When a test utterance is given, it can compute the likelihood for each cluster and make a comparison. Cluster with the highest probability means the corresponding speaker. In GMM, data are modeled as different clusters, each has its own mean vector, weight parameter, and covariance matrix. It can be represented as

$$f(x_n|\lambda) = \sum_{i=1}^{M} \pi_i \mathcal{N}(x_n|\mu_i, \Sigma_i), \tag{1.1}$$

where n is the index for a random vector, i is the index for the cluster while M is the number of clusters. μ_i is the mean vector and Σ_i is the covariance matrix. π_i is the weight for each component of the GMM. The likelihood of an utterance is given by Eq. 1.1. If we denote the GMM by $\lambda = \{\pi_i, \mu_i, \Sigma_i | i = 1, 2, ..., M\}$, the we can use multiplication of

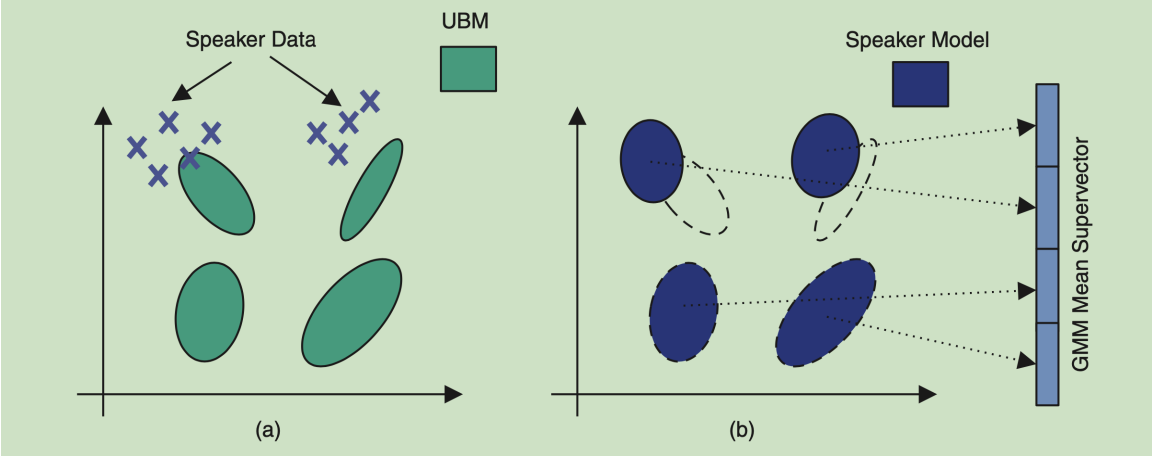

Figure 1.2: Diagram of GMM-UBM mixture model. (a) GMM–UBM system with four-mixture UBM. (b)MAP adaptation and supervector formation by concatenating the mean vectors.

probabilities to represent the likelihood of a series of features $\mathcal{X} = \{x_1, x_2, ..., x_N\}$ as

$$p(\mathcal{X}|\lambda) = \prod_{i=1}^{N} p(x_i|\lambda). \tag{1.2}$$

With the help of GMMs, and to make the speaker verification more general, universal background model (UBM) was proposed [12], in which the speaker model can be modified based on a background or world model. A UBM acts as a model for all enrolled speakers, so before the training of the speaker model, a UBM can be taken as the initial status for the speaker model. This combination is called the GMM-UBM method. Bayesian adaptation is performed on a speaker's GMM. In GMM-UBM, a supervector made by concatenating the parameters of each component is used as the feature vector for verification, which is shown in Fig. 1.2 [6]. The GMM supervector can be used with a support vector machine (SVM) [13] which can use a hyperplane to do the classification and can maximize the margin between two datasets.

The GMM supervector can be viewed as a linear combination of several components, a speaker/channel/environment-independent component ($\mathbf{v}_0$), a speaker-dependent component ($\mathbf{v}_{spk}$), a channel/environment-dependent component ($\mathbf{v}_{ch}$) and a residual component

($\mathbf{r}$), which can be represented as

$$\mathbf{v}_{spk,ch} = \mathbf{v}_0 + \mathbf{v}_{spk} + \mathbf{v}_{ch} + \mathbf{r}. \tag{1.3}$$

However, the dimension of supervector $\mathbf{v}$ is huge so several lower dimensional representations are proposed. For example, classical maximum a posterior probability (MAP) adaptation uses

$$\mathbf{v}_{spk} = \mathbf{v}_0 + \mathbf{D}\mathbf{z}_{spk}, \tag{1.4}$$

where $\mathbf{D}$ is a diagonal matrix and $\mathbf{z}_{spk}$ is a standard normal random vector. Also, another speaker-based eigenvoice adaptation was proposed [14] as

$$\mathbf{v}_{spk} = \mathbf{v}_0 + \mathbf{M}\mathbf{y}_{spk}, \tag{1.5}$$

where columns of matrix $\mathbf{M}$ spans the speaker subspace. In this way, $\mathbf{y}_{spk}$ can be used as coefficient for the basis vectors of the speaker space. Similarly, eigenchannel was proposed in [15] as

$$\mathbf{v}_{spk,ch} = \mathbf{v}_0 + \mathbf{D}\mathbf{z}_{spk} + \mathbf{U}\mathbf{m}_{ch}, \tag{1.6}$$

where $\mathbf{U}$ spans the subspace for channel/environment. So the joint factor analysis (JFA) was formulated by combining both eigenspeaker and eigenchannel. This model can take the speaker and channel information into consideration at the same time, outperforming other FA methods [16], which can be represented as

$$\mathbf{v}_{spk,ch} = \mathbf{v}_0 + \mathbf{U}\mathbf{x}_{spk} + \mathbf{V}\mathbf{y}_{ch} + \mathbf{M}\mathbf{z}_{spk,ch}. \tag{1.7}$$

GMM with SVM classifiers perform very well on the speaker verification task. FA based methods also have good performance. In order to combine the advantages of the

two approaches, [17] proposed to use JFA as feature extractor for SVMs. In the related work, channel and speaker information are merged into one total variability space. In the FA model, a speaker and session dependent GMM supervector is represented by

$$\mathbf{v}_{spk,ch} = \mathbf{v}_0 + \mathbf{T}\mathbf{w}_{spk,ch}, \tag{1.8}$$

where hidden variables $\mathbf{w}_{spk,ch} \sim \mathcal{N}(0, \mathbf{I})$ and is called total factors. Similar to JFA and FA methods, hidden variables cannot be observed but can be estimated by the posterior expectation. The estimate of the total factor is called identity vector, which is also called i-vector, can be used as features for classifiers.

However, the i-vector approach does not distinguish between speaker information and channel information. It is just a dimensionality reduction of GMM supervector, which is like principal component analysis (PCA) on GMM supervectors. Matrix $\mathbf{T}$ is trained similarly to that in eigenvoice model.

With the application of machine learning and deep learning, from recent research, [18] focuses on DNN-based text-dependent verification with "Hello Google" and performs well. In [19] the focus is more on text-independent verification and is used as a baseline in many papers in the ASV field. In this thesis, x-vectors are used for ASV and technical details can be found in Section 2.1.8.

1.1.4 Back-End Approaches of ASV

Usually we split ASV into two parts, the front-end method and the back-end method. The front-end method is the extraction of high-level information of the utterance or speech. The back-end method is the classifier or the decision making model. In Section 1.1.3, front-end methods from GMM modeling to i-vector extraction are introduced. In this section, back-end methods will be introduced.

First linear discriminant analysis (LDA) is introduced. It is a method widely used in statistics, pattern recognition, and machine learning to find a linear combination of features

that can separate two or more classes. It finds the orthogonal directions in the feature space that are more effective in discriminating the classes. The projection from the original vector to the LDA dimensions can improve the classification accuracy. If the set of all training audio is denoted by D, $\mathbf{w}_{s,i}$ denotes a feature of audio from the i-th audio file from speaker s. n_s denotes the number of audio files for speaker s, and the total number of all audio files in D is denoted as S. Then the inter-class and intra-class covariance matrices are computed by

$$\mathbf{C}_{inter} = \frac{1}{S}\sum_{s=1}^{S}(\overline{\mathbf{w}}_s - \overline{\mathbf{w}})(\overline{\mathbf{w}}_s - \overline{\mathbf{w}})^T, \tag{1.9}$$

and

$$\mathbf{C}_{intra} = \frac{1}{S}\sum_{s=1}^{S}\frac{1}{n_s}\sum_{i=1}^{n_s}(\mathbf{w}_{s,i} - \overline{\mathbf{w}}_s)(\mathbf{w}_{s,i} - \overline{\mathbf{w}}_s)^T, \tag{1.10}$$

in which the speaker-independent and the speaker-dependent mean vectors are respectively denoted by

$$\overline{\mathbf{w}}_s = \frac{1}{n_s}\sum_{i=1}^{n_s}\mathbf{w}_{s,i}, \tag{1.11}$$

and

$$\overline{\mathbf{w}} = \frac{1}{S}\sum_{s=1}^{S}\frac{1}{n_s}\sum_{i=1}^{n_s}\mathbf{w}_{s,i}. \tag{1.12}$$

In LDA, the optimization is aimed at minimizing the within-class variance and maximizing the between-class variance. The projections can be obtained by solving $\mathbf{v}$ for the equation

$$\mathbf{C}_{inter}\mathbf{v} = \mathbf{W}\mathbf{C}_{intra}\mathbf{v}, \tag{1.13}$$

where $\mathbf{W}$ is a diagonal matrix containing the eigenvalues for $\mathbf{C}_{intra}^{-1}\mathbf{C}_{inter}$. When $\mathbf{C}_{intra}$ is invertible, we can compute the eigenvalue of matrix $\mathbf{C}_{intra}^{-1}\mathbf{C}_{inter}$. To compute the utterance

feature, we can use first k eigenvalues for the LDA matrix as

$$\mathbf{M}_{LDA} = [\mathbf{v}_1, ..., \mathbf{v}_k], \tag{1.14}$$

where $\mathbf{v}_1, ..., \mathbf{v}_k$ are first k eigenvectors of $\mathbf{C}_{intra}^{-1}\mathbf{C}_{inter}$. Thus, the feature $\mathbf{w}$ can be computed as

$$\Phi_{LDA}(\mathbf{w}) = \mathbf{M}_{LDA}^{T}\mathbf{w}. \tag{1.15}$$

After the LDA computation, there are several methods for the classification of speaker identity. First, people used SVM classifier after JFA feature extractions. Then, cosine distance scoring was introduced in [20]. In this case, the matching score between test feature and target feature is computed as

$$Score(\mathbf{w}_{target}, \mathbf{w}_{test}) = \frac{\mathbf{w}_{target} \cdot \mathbf{w}_{test}}{\|\mathbf{w}_{target}\| \, \|\mathbf{w}_{test}\|}. \tag{1.16}$$

After this, people began to use probabilistic LDA (PLDA) [21], which uses the pipeline of JFA. In PLDA, the i-vector can be decomposed as

$$\mathbf{w}_{spk,ch} = \mathbf{w}_0 + \boldsymbol{\Theta}\boldsymbol{\alpha}_{spk} + \boldsymbol{\Upsilon}\boldsymbol{\beta}_{ch} + \boldsymbol{\varepsilon}_{spk,ch}, \tag{1.17}$$

where $\mathbf{w}_0 \in \mathbf{R}^R$ is the speaker-independent mean i-vector. $\boldsymbol{\Theta}$ is $R \times N_{eigenspk}$ low-rank matrix for speaker-dependent basis functions. $\boldsymbol{\Upsilon}$ is $R \times N_{eigench}$ low-rank matrix for the channel space. $\boldsymbol{\alpha}_{spk} \in \mathcal{N}(0, \mathbf{I})$ is a $N_{eigenspk} \times 1$ hidden variable representing speaker factor, $\boldsymbol{\beta}_{ch} \in \mathcal{N}(0, \mathbf{I})$ is a $N_{eigench} \times 1$ hidden variable representing channel factor, and $\boldsymbol{\varepsilon}_{spk,ch} \in \mathbf{R}^R$ is a random vector denoting residual noise. Also, a full-covariance noise model $\boldsymbol{\varepsilon}_{spk,ch}$ can be adopted to drop the channel component in Eq. 1.17, which yields

$$\mathbf{w}_{spk,ch} = \mathbf{w}_0 + \boldsymbol{\Theta}\boldsymbol{\alpha}_{spk} + \boldsymbol{\varepsilon}_{spk,ch}. \tag{1.18}$$

1.1.5 Performance Metric

Equal error rate (EER) is usually the metric to measure the performance of the ASV system. In order to describe EER, false acceptance rate (FAR) and false rejection rate (FRR) should be introduced first.

$$\begin{aligned} \text{FAR} &= \frac{\text{\#False Acceptance}}{\text{\#Verification Attempts}}, \\ \text{FRR} &= \frac{\text{\#False Rejection}}{\text{\#Verification Attempts}}. \end{aligned} \tag{1.19}$$

FAR results from accepting the claim of non-target speaker while FRR results from declining the claim of the target speaker. Usually in real applications, people can design their ASV system to meet their own demand for the performance system. For applications such as banking verification, people tend to lower the FAR while increasing the FRR. In applications such as unlocking the cellphone, people can lower the FRR while increasing the FAR. In this thesis, EER will be the metric for analyzing the cleanness of each dataset, and is defined as the point when FAR is equal to FRR.

1.2 VoxCeleb Data Collection

VoxCeleb [5, 22] is an audio-visual dataset consisting of short clips of human speech, extracted from interview videos uploaded to YouTube. There are 1251 speakers with 153516 utterances in VoxCeleb 1 and 6112 speakers with over one million utterances in VoxCeleb 2. The pipeline of VoxCeleb is shown in Fig. 1.3 [5]. The main language of VoxCeleb is English, with 37% of speakers from the USA, the others are from the UK, France, India, Germany, etc. In VoxCeleb, 61% of speakers are male speakers while the remaining 39% are female speakers.

From Fig. 1.3, data collection procedure includes five stages: Preparing list of persons of interests (POIs), downloading videos from YouTube, face tracking, active speaker verification, and face verification. Details of our implementation and modification are elaborated

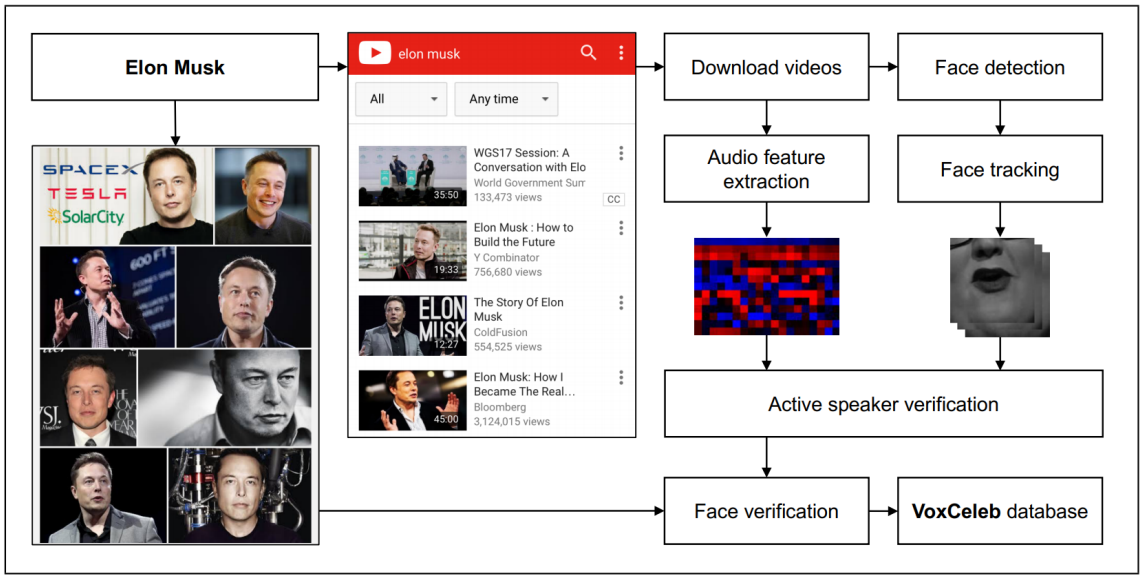

Figure 1.3: VoxCeleb data processing pipeline.

in Section. 2.1.

1.3 Speaker Diarization

Speaker diarization is the process of partitioning an input audio stream into homogeneous segments according to the speaker identity [23]. It can enhance the readability of an automatic speech transcription by structuring the audio stream into speaker turns and, when used together with speaker recognition systems, by providing the speaker's true identity. So basically it is a task to answer "Who spoke when." Speaker diarization essentially includes speaker clustering and speaker segmentation. The former part is to determine different regions of each speaker and the second part is to group together speech segments depending on the characteristics. The function of speaker diarization is shown in Fig. 1.4 [23]. In this case, diarization is done on a broadcast news and gray parts of the recording are classified as commercial news. Brown, yellow, light blue, and pink parts are classified as different speakers. Also some crowd noise denoted by white are also classified by the diarization system.

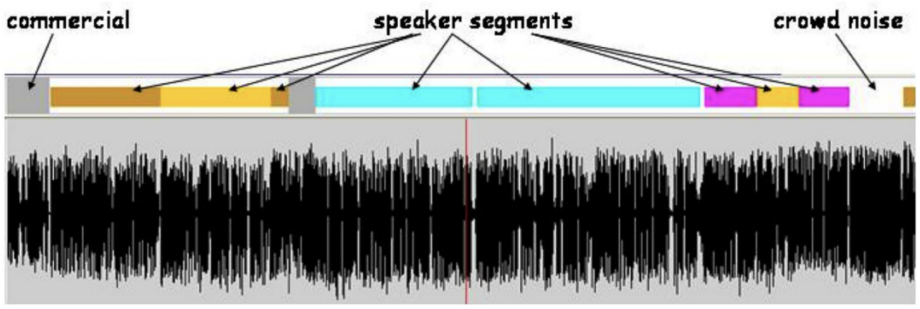

Figure 1.4: Audio diarization on broadcast news.

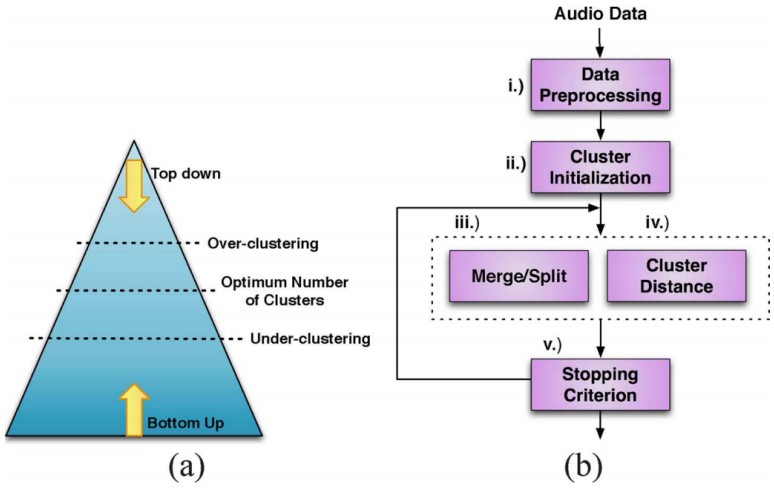

Figure 1.5: Diarization system. (a) Top-down and bottom-up approaches. (b) General architecture.

Generally speaking, there are two main diarization methods, bottom-up and top-down, which are shown in Fig. 1.5 [24]. The top-down approach starts with usually one cluster and splits after steps according to a distance criterion, while the bottom-up approach starts with usually a number of segments and merges some of them together after steps. Both of them iterate until reaching the number of optimum number of clusters.

In this thesis, diarization is used as a tool to filter the noisy speakers in the proposed pipeline, details of our approach are elaborated in Section 2.1.7.

CHAPTER 2
TECHNICAL APPROACH

In this section, our proposed data collection pipeline is introduced, then the methods for testing our data are also introduced.

2.1 Proposed Pipeline

In our research, work is based on data collection of non-English speaking celebrities, while adding speaker diarization to the pipeline of VoxCeleb in order to make the dataset cleaner. Our data collection pipeline is shown in Fig. 2.1. Most of the parts are similar to that of VoxCeleb while in speaker diarization part, we want to use it as a filtering approach to remove the noisy segments of the VoxCeleb output instead of human verification. Our detailed plan is described as follow.

The reason we select Asian speakers as our source is that, in VoxCeleb the main language is English. Usually for different languages, the characteristics may change. So it would be beneficial to add more languages to the current available speaker recognition/verification datasets. Including more languages may help the model trained on the dataset to be more robust to different test cases, which is the aim of this work.

2.1.1 Collecting List of POIs

In this work, the focus is put on non-English speaking celebrities mainly in east Asia because of the relatively large difference between Asian languages with English. This part focuses on collecting persons of interests (POIs). Our collected dataset is named as east Asian celebrities (EACeleb), consisting of Chinese, South Korean, Japanese celebrities. Names of POIs are collected from Forbes top 100 list, Wikipedia, and some native ranking websites with searching keywords such as actor, singer, player, etc in the respective native

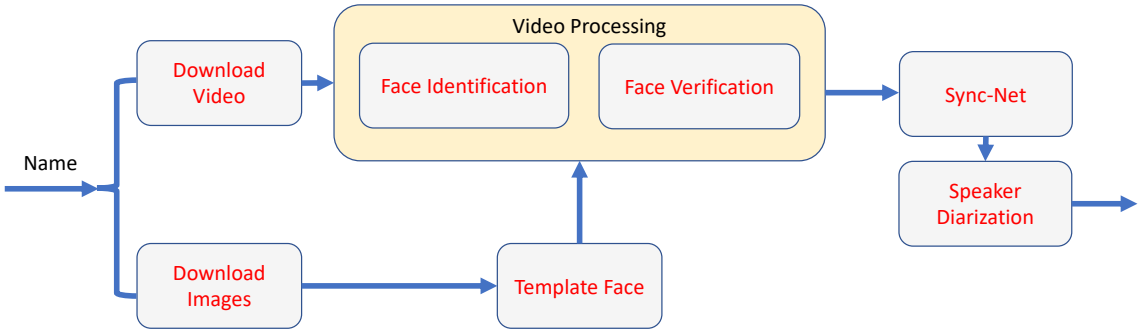

Figure 2.1: Proposed data collection procedure.

Table 2.1: Statistics of collected speakers.

Language	Number of Speakers
Chinese	410
Japanese	406
Korean	925
Total	1741

languages. The numbers of speakers are shown in Table 2.1. Actually, the initial list of speakers was larger than the number shown in the table, which are 510 for Chinese speakers, 847 for Japanese speakers, and 1633 for Korean speakers. However, some speakers lack sufficient data available on YouTube so not all of the speakers in the original list can be used to make up the EACeleb. Detailed information about the names of speakers can be found in Appendix B.

2.1.2 Creating Template Face

In this stage, we searched the name of the specific celebrity in their own language with the word "photo" appended. Based on the origin of the celebrity, the word "photo" was substituted with the Google translation into the target language. We used Google Images Download library to download 25 images for the target speaker. Since the downloaded images may not be all the celebrity, for the downloaded images, we first performed face detection on all images using face detector in Dlib library, and then clustered the results to

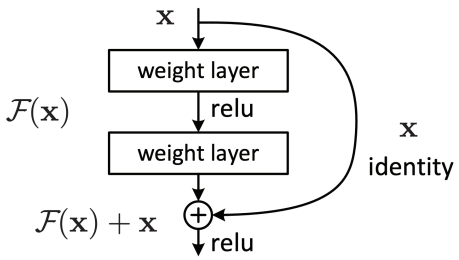

Figure 2.2: Building block for ResNet.

make sure the output is our target speaker.

For face embedding, we use VGGFace from the Keras toolkit to extract an embedding for each face image. For the VGGFace, the backbone model we used is ResNet50 [25] and the input size was reshaped into $224 \times 224 \times 3$. The basic module of ResNet is shown in Fig. 2.2 [25], which can transmit the residual of next module to the module before and prevent the residual fading problem. The output dimension is 2622. By doing this, we could get the face embedding for each face in the downloaded images.

For face clustering, we used density-based spatial clustering of applications with noise (DBSCAN) to create a template face representing the target celebrity. Here the threshold ϵ was set to be 0.4, if the value of cosine distance function between two embeddings was larger than ϵ then they cannot be put together. After clustering we choose the largest cluster as our template face and saved the embedding for comparison in our pipeline later.

2.1.3 Downloading Videos from YouTube

In this stage, we downloaded top 20 videos of each of the POIs on YouTube. Due to limited hardware, we could not download 50 videos as with VoxCeleb. The translation of the word "interview" to the original language was appended to the name of the celebrity in their own languages.

In our approach, we copied the url of top 20 video results and used youtube-dl, which is a command-line program, to download the videos. Each video was stored in a directory with the name of its YouTube id as was done for VoxCeleb. After downloading all videos,

we checked the total length of the video. If the length was less than 1000 seconds, we skipped that video.

2.1.4 Face Detection

In each frame, we did face detection to compare the person of the frame with our template face of POI. In our approach, we used the Haar-based face detection. First, RGB images of faces were changed into grayscale image. Then we extracted Haar features. Haar-like features are digital image features used in object recognition. The name is for their intuitive similarity with Haar wavelets and was used in the first real-time face detector. These features utilize the face that the eyes are darker than regions of cheeks and are darker than the bridge of noses. For the classifier we used adaptive boosting (AdaBoost).

2.1.5 Face Verification and Tracking

In this step we compared the face of the video frame with the template face of the target celebrity. First, we extracted the feature for the detected face using ResNet as we did in Section 2.1.2, then compared it with the template face using a cosine similarity score. In our approach, if the similarity score is over 0.6, then it's declared a match and the face in the current frame is our target celebrity.

In addition to the face verification, we also tracked the face when the detected face was from our template face of the target speaker. Face tracking was implemented using the CSRT tracker from OpenCV. To make sure that the face was tracked properly, the histogram of the face was checked in each iteration. The minimum intersection over union (IoU) between consecutive detected face was set to be 0.5. In each track process, three times of missed detection were allowed.

To help the face tracking, scene detection was also adopted because for an interview, when the scene changes, it usually also signifies a people switch and speaker change. We used the Python pyscenedetect package and used the color histogram-based scene detection

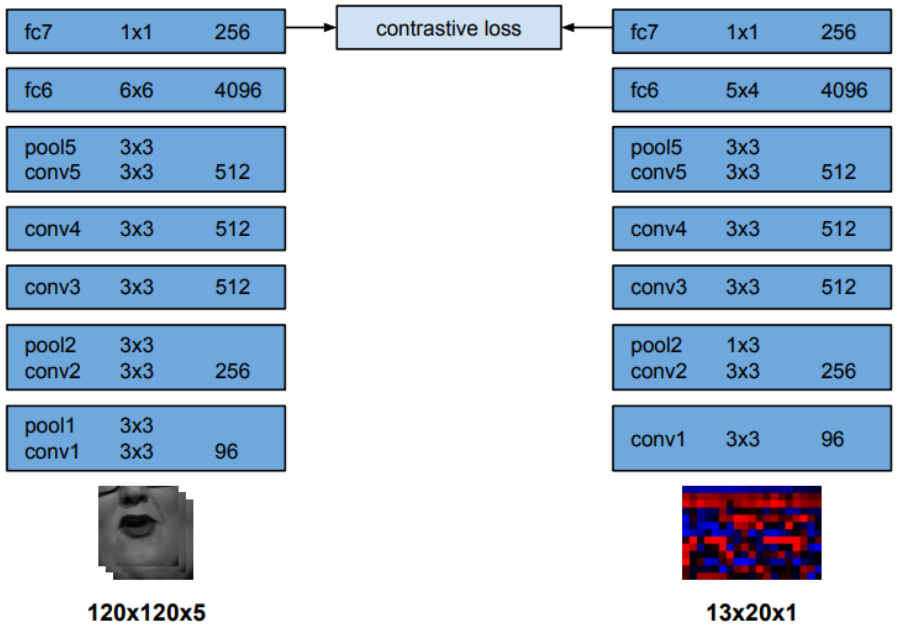

Figure 2.3: Two-stream ConvNet architecture. Both streams are trained simultaneously.

algorithm in HSV/HSL color space and store the frames where the scenes transitioned. By doing this, we could first divide the whole video into specific segments consisting of several scenes.

After the face tracking process, we obtained several segments of video from the YouTube video. Then audio tracks of the video were also extracted and the cropped segment of the original video was saved.

2.1.6 Audio Visual Synchronization

In this step, we used an audio-to-video synchronization network (Sync-Net) [26] to synchronize the audio and lip movement. It can remove temporal lags between the audio and visual streams in a video, and determine who is speaking among multiple faces in a video. It takes the MFCCs from audio as the feature.

The model was trained simultaneously, one stream for images in the video frames and one for audio, as shown in Fig. 2.3 [26].

2.1.7 Speaker Diarization

In this step, speaker diarization was done to filter out the noisy part of the collected data. In this case, we used a state-of-the-art front-end method instead of the conventional i-vector approach. With the advancement of neural networks in recent years, machine learning, and deep learning have been used to solve many problems in different areas. Convolutional neural networks (CNNs) are usually used for image classification or tasks relating to images. Deep neural networks (DNNs) are usually used for multi-class classification problems and recurrent neural networks (RNNs) are used for series processing. In the field of ASV, [19] proposed a time-delayed neural network (TDNN)-based method named x-vector for speaker embedding and proved its advantage over traditional i-vector method. In our approach, x-vector is extracted for PLDA scoring.

A diagram of x-vector extraction is shown in Fig. 2.4 [27]. From the diagram we can see that x-vector extracts the frame-level information, then the statistics pooling layer aggregates the frame-level information, and the segment-level processes the representation after this. Finally, there is a softmax output layer for the speaker classification. The activation function is rectified linear units (ReLUs).

In the x-vector scheme, the first 5 layers work at the frame level with a time-delayed architecture [27]. For example, if t is the current time step, the input frames of $\{t-2, t-1, t, t+1, t+2\}$ are included in computation. The next two layers are the output of the previous at $\{t-2, t, t+2\}$ and $\{t-3, t, t+3\}$. The following two layers just operate at the frame-level representations and no temporal context are added. So in total, context from $t-8$ to $t+8$ frames are included. The size of layer varies depending on the context used. Afterwards, statistics polling layer aggregates over the final output of the frame-level representations and computes the mean and standard deviation. These statistics are concatenated together and two additional segment-level layers are added after the pooling layer. The output layer is a softmax layer for classification problem and after training the whole neural network, two segment-level layers can be used for the speaker embedding.

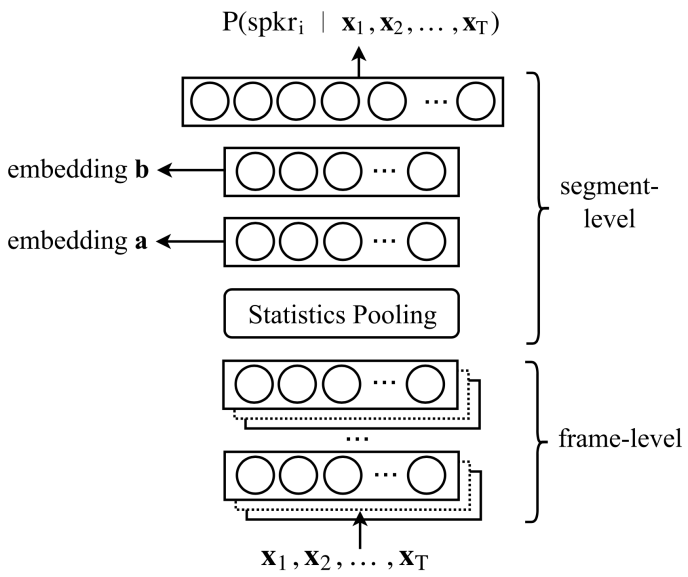

Figure 2.4: Diagram of x-vector extraction.

In our approach, we used pretrained SRE16 and VoxCeleb x-vector models from Kaldi for x-vector extraction in speaker diarization part. Then we modified the script of the Kaldi callhome recipe to do the diarization. Details of our experiments can be found in Section 2.2.1.

2.1.8 Speaker Verification

To measure the cleanness of EACeleb and EACeleb after diarization, we used ASV to compute the EER on the dataset. Because we lack ground truth about each speaker and the corresponding segments, we assumed that each audio segment from the final output of our pipeline was from the target speaker. By this assumption, if we used ASV for the testing, the lower EER is, the cleaner the dataset should be. In our approach, we decided to measure the dataset in two ways as follows.

The first way was to load the pretrained model that trained on a dataset other than EACeleb and test the speakers in EACeleb for the EER computation. In this case, we use SRE16, SITW, and VoxCeleb x-vector models from Kaldi models.

The second way was to train our own model based on the original EACeleb and EACeleb

after diarization, and test the model using the testing data from VoxCeleb, original EACeleb, and EACeleb after diarization. Model details of the experiments are shown in Section 2.2.2.

2.2 Experiment Setup

In this section, experimental details of speaker diarization and speaker verification are introduced.

2.2.1 Speaker Diarization

In our approach to diarization, we modified the Kaldi Callhome diarization script on NIST SRE 2000 to adjust to our EACeleb. We used two different x-vector models, which are Kaldi SRE16 and VoxCeleb model.

First, we prepared the wav.scp, spk2utt, and utt2spk file needed by Kaldi for processing. The wav.scp file contains the information of the identity for the speaker and the path to the audio file, with the channel information and coding information. The spk2utt and utt2spk files are the relationship of the audio file and the speaker identity for the file.

After the basic file preparation, MFCC features are extracted from the audio file. In diarization, the whole audio track of the output video is used and downsampled to 8 kHz for processing. In this case, frame length is set to be 25 ms, the lowest frequency is 20 Hz while the highest frequency is 3700 Hz. Then, voice activity detection (VAD) is done to each frame, energy threshold of VAD is set to be 5.5, energy mean scale is set to be 0.5, proportion threshold is set to be 0.12 while frame contexts is set to be 2. After VAD, voice parts are separated and stored as segments for x-vector extraction. The configuration of the x-vector model is not changed according to the Kaldi pretrained model. After extracting x-vectors of each utterance, corresponding PLDA scores are computed and stored. Finally, we set the number of speakers in a recording to be 3 and do supervised diarization. The reason for setting 3 is that, we assume there would be one target speaker in our output. For the other two, one is for the interviewer or audience from TV show, the other is some noisy

speaker, such as speaker with background noise, a speaker speaking in an abnormal manner (e.g. singing), etc.

Diarization produces an rttm file showing the start time and end time for each assigned speaker in a specific recording. Using the file, we wrote a bash script to process the original EACeleb. The first step is to determine the dominant speaker of each recording using the total length among three speakers; we selected the speaker with the longest duration as the dominant speaker. Next we concatenated the dominant speaker parts in the original output using corresponding start and end times to create a new dataset as EACeleb after diarization. Results and statistics of EACeleb after diarization are shown in Section 3.1.

2.2.2 Speaker Verification

The original EACeleb and EACeleb after diarization were analyzed in terms of EER. First, audio track of the videos in these datasets were extracted and downsampled to 8 kHz for processing. Then they were split into 7, 10, 13, 15 seconds segments for testing purposes. The corresponding numbers of utterances in each case are shown in Table 2.2. Then corresponding wav.scp, spk2utt, and utt2spk files were generated afterwards.

Table 2.2: Number of utterances for each split of datasets.

Dataset \ Duration	7s	10s	13s	15s
EACeleb	148976	104558	80622	69996
EACeleb after SRE Diarization	141378	99218	76539	66435
EACeleb after VoxCeleb Diarization	142660	100124	77214	67057

For speaker verification, we used two ways of testing as described in Section 2.1.8. Firstly, using the pretrained model approach is introduced. Then follows the training-our-own-model approach.

In order for a fair comparison, we used Kaldi VoxCeleb, SRE16, and SITW models for testing. In this case, we follow the usual method of ASV. First extract MFCC feature of each segment, then perform the VAD. Afterwards we extract the x-vector and do the PLDA

scoring, finally compute the EER result. For testing purposes, the datasets we use are the original EACeleb, EACeleb after SRE diarization, and EACeleb afer VoxCeleb diarization, in which SRE diarization denotes the output of speaker diarization using x-vector model of SRE in Kaldi on EACeleb. Also, the testing results of VoxCeleb testing data are also included. In this thesis, VoxCeleb testing data means all audio files from VoxCeleb2 test set. However, there are some slight differences between VoxCeleb and our testing datasets.

In our testing experiments, we randomly select 1591 speakers for development data and use the remaining 150 speakers for enrollment data and testing data. For EACeleb, EACeleb after SRE diarization, and EACeleb after VoxCeleb diarization datasets, variables such as mean vector, LDA, and PLDA are computed based on the development data. For enrollment, the first utterance of each speaker in testing data are selected as enrollment data. Then in testing, all utterances other than the first utterance from the test speakers are used as testing data, and here each utterance is viewed from single speaker. In the trial file, utterances from the same speaker are viewed as target trial and we compared one utterance in testing data with all other utterances. For VoxCeleb testing data, because it's divided into many folders, we randomly select one audio file for each speaker as enrollment audio and use all other audio files as testing data. Because VoxCeleb2 testing audio files don't have a fixed duration, so we use "vary" in the duration of VoxCeleb. The mean vector, LDA, and PLDA transform are all from the pretrained VoxCeleb model.

For training our own data, we trained the x-vector model based on the development data using the default x-vector configuration in Kaldi, then test the model using the same approach above. In this case, we trained models for original EACeleb, EACeleb after SRE diarization, and EACeleb after VoxCeleb diarization. Then test the models to all other testing datasets. In this step, we only use the datasets with 10-second durations. The results of two ways above are shown in Section 3.2.

CHAPTER 3

RESULTS

In this section, results of speaker diarization and verification are elaborated and analyzed. Speaker diarization results are shown in Section 3.1 and EER results of testing and our own training are shown in Section 3.2.

Before speaker diarization, three lists of celebrities speaking the corresponding language are collected from Forbes top list, wikipedia, or from google searches. We run the script to collect original video first and the numbers of speakers we collected are shown in Table 2.1. The total duration of the dataset is about 290 hours.

Next, ffmpeg is used to extract the audio track of the collected video. Then we downsample the audio file from 16 kHz to 8 kHz using sox. Now we have the downsampled audio for each speaker. Then we do the speaker diarization on all audio files we collected to make the dataset cleaner.

3.1 Speaker Diarization Results

For the speaker diarization, based on the Kaldi [28] callhome diarization v2 recipe, we modify the script on our dataset. We used the SRE16 and VoxCeleb x-vector model from Kaldi to extract the x-vector of our own dataset. The steps are: extracting MFCC feature, extracting x-vector, PLDA scoring and diarization. In the diarization step, we used the supervised diarization which means we need to provide a file with total number of speakers in an audio file. In our dataset, because the pipeline of VoxCeleb can get rid of most of the noise, we set the number of speaker in an audio to be 3, one of them is the target speaker and should be the main part of the audio. The second is the noisy speaker such as the interviewer, which is not the speaker we want. The third is assumed to be an unknown speaker with noise such as audience speaking, the target speaker speaking in abnormal

Table 3.1: Useful collection ratio of collected speakers in percentage (SRE diarization).

Language	UCR after Diarization
Chinese	93.82
Japanese	87.01
Korean	87.88
Average	89.08

Table 3.2: Useful collection ratio of collected speakers in percentage (VoxCeleb diarization).

Language	UCR after Diarization
Chinese	94.50
Japanese	88.40
Korean	87.31
Average	89.26

sound, etc. Here we define the useful collection rate (UCR) of a data collection, which means the ratio of the duration of the audio after diarization to the duration of the original audio. Corresponding results are shown in Table 3.1 and Table 3.2.

$$\text{UCR} = \frac{\text{Duration of audio after diarization}}{\text{Duration of original audio}}. \tag{3.1}$$

To have a better sense of what speaker diarization did on the original EACeleb, we separate the original part and the discarded part. Here three typical example speakers are discussed. First is Dehua Liu or Andy Lau, a famous actor in China. In the collection output of him, most parts are from interview scenes. While in the original EACeleb, in addition to what is shown in Fig. 3.1, which correspond with the face of the target speaker Andy Lau, there exists another person speaking, as shown in Fig. 3.2. Watching the discarded part, most of it contains the video from the interviewer and for some segments Andy's face is shown while he is not speaking.

The second example is from Beining Sa, a TV host. The video output of him is already clean enough from the video output as shown in Fig. 3.3. In the original EACeleb of Beining Sa, there is only the target speaker in the video. While after diarization, in the

Figure 3.1: Screenshots of the video output for Andy Lau.

Figure 3.2: Screenshots of discarded video output for Andy Lau.

Figure 3.3: Screenshots of video output for Beining Sa.

Figure 3.4: Screenshots of discarded video output for Beining Sa.

discarded part as shown in Fig. 3.4, most of it are from him singing in a TV show.

For the third speaker Na Li, a famous Chinese tennis player, the video output of her in original EACeleb is clean in terms of face. Almost no noisy speaker exists in the video output, screenshots of the video are shown in Fig. 3.5. After diarization, screenshots of discarded part are shown in Fig. 3.6. In this case, discarded parts are mostly frames that Na Li is speaking with background music, a few frames of other noisy speakers, and some frames she appears while the sound is from the commentator.

3.2 Speaker Verification Results

Since the pipeline cannot determine whether the video/audio we collected and selected after diarization are truly from our target speaker, we have to use speaker verification method to verify our data by EER. Speaker verification can determine if some part of the audio is


Figure 3.5: Screenshots of video output for Na Li.


Figure 3.6: Screenshots of discarded video output for Na Li.

from other person because the voiceprints of different people are different and the speaker verification technique nowadays are very percise. To make our comparison more general, we have two ways of measuring the data cleanness. First is to measure our data by existing x-vector models and compare the EER of the data after diarization with the original data. Second is to train the x-vector model using our collected data and measure the VoxCeleb data in terms of EER. Results are shown in sections below.

3.2.1 Testing Our Collected Data

In order to test our data, we have to load trained x-vector models for feature extraction. In our experiment we used VoxCeleb, SITW, and SRE16 model from Kaldi Models. Speaker verification is based on Kaldi sre16 v2 recipe. Experiments are done on original collected data and data after diarization. Our approach to generate trial files is described in Section 2.2.2. The EER values are shown in Table 3.3.

As can be seen from the table, the x-vector models we use are VoxCeleb, SITW, SRE16. We compared three datasets which are original EACeleb, EACeleb after speaker diarization using SRE x-vector model and EACeleb after diarization using VoxCeleb x-vector model.

First, we analyze the results between original EACeleb and EACeleb after SRE diarization. For VoxCeleb testing model on 10 second segments, speaker diarization can lower the EER of our collected data by 18.42%. For SITW model of 10 second segments, speaker diarization can lower the EER of our data by 14.86% and outperform VoxCeleb by 45.69%. For SRE16 model on 10 second segments, speaker diarization can lower the EER of our data by 25.63% and outperform VoxCeleb by 49.12%. Then, for 13 second segments, SRE diarization can improve the EER by 23.19 on VoxCeleb model. For all cases in our testing, speaker diarization can improve the EER performance of EACeleb, which shows that speaker diarization is feasible and useful in denoising the output of VoxCeleb data collection.

Second, differences between EACeleb and EACeleb after VoxCeleb diarization are discussed. Actually speaker diarization using the VoxCeleb can still outperform all cases in our experiments of EACeleb. For 15 second segments using the VoxCeleb model, VoxCeleb diarization can improve EER of EACeleb by 11.58%. For 7 second segments using SITW model, VoxCeleb diarization can achieve 10.46% EER reduction compared with EACeleb. From the analysis above, we can see that speaker diarization using different models can both improve the EER performance of the datasets after diarization under ASV measurement.

Next, we will analyze the difference between EACeleb after SRE diarization and EACeleb after VoxCeleb diarization. For 13 second segments using the SRE16 model, EACeleb after SRE diarization can outperform EACeleb after VoxCeleb diarization by 23.37%. For 10 second segments under VoxCeleb model, EACeleb after SRE diarization has 16.77% EER reduction compared with EACeleb after VoxCeleb diarization. Also, all results in our experiments show that EACeleb after SRE diarization is cleaner than EACeleb after Vox-

Table 3.3: EER performances of collected data using different models (in percentage).

Dataset	EACeleb				EACeleb after Diarization (SRE)				EACeleb after Diarization (VoxCeleb)				VoxCeleb
Duration / Model	7s	10s	13s	15s	7s	10s	13s	15s	7s	10s	13s	15s	Vary
VoxCeleb	21.789	19.946	17.640	17.260	18.881	16.272	13.548	12.706	20.358	19.550	15.742	15.621	27.282
SRE16	19.062	17.101	16.851	16.256	14.983	12.718	11.351	11.012	16.819	16.739	15.008	14.441	24.996
SITW	21.281	19.037	17.285	16.712	18.576	16.209	13.181	12.753	18.982	17.895	14.927	14.771	29.597

Table 3.4: EER performances of our trained models (in percentage).

Model / Dataset	EACeleb	EACeleb after Diarization (SRE)	EACeleb after Diarization (VoxCeleb)	VoxCeleb
EACeleb	22.669	17.830	21.049	28.017
EACeleb after Diarization (SRE)	21.277	17.363	21.236	27.465
EACeleb after Diarization (VoxCeleb)	21.398	17.861	21.174	29.080

Celeb diarization because EER are lower than the corresponding value in VoxCeleb. The reason is that SRE model is trained on NIST SRE dataset, which cleaner than VoxCeleb. So the SRE x-vector model can extract speaker characteristics better than the VoxCeleb one.

It's also worth noting that in this thesis, we used a different way of enrolling and testing speakers from VoxCeleb, so that's why the EER in the thesis is much higher than in the VoxCeleb paper [5, 22]. In our way of testing, we can see from the EER results that, all EER results for EACeleb and EACeleb after diarization are lower than the results of VoxCeleb, which means the single utterance of speakers in EACeleb, EACeleb after SRE diarization, and EACeleb after VoxCeleb diarization, can better represent the speaker in general.

3.2.2 Training EACeleb Models

For the collected data, we used all utterances from 1591 speaker in development data and the remaining 150 as testing speakers. Also to facilitate comparison, results of testing our trained model using VoxCeleb testing data are also included. The results are shown in Table 3.4.

In our approach, we trained three models based on speakers in development data for original EACeleb, EACeleb after SRE diarization, and EACeleb after VoxCeleb diarization, respectively. In our experiments, testing data are the testing data from original EACeleb, EACeleb after SRE diarization, EACeleb after VoxCeleb diarization, and VoxCeleb, respectively.

First, for EACeleb model, testing results of EACeleb after SRE diariation show the best performance in this case, which has a 21.35% advantage over EACeleb testing data while outperforming VoxCeleb by 36.36%. Also EER testing EACeleb after VoxCeleb diarization has 15.29% higher EER than EACeleb after SRE diarization.

Then, using the model trained on EACeleb after SRE diarization, EER performance of VoxCeleb testing data is 27.465% and it is outperformed by all cases of EACeleb; The

original EACeleb can outperform VoxCeleb by 22.53%, EACeleb after VoxCeleb diarization can outperform VoxCeleb by 22.68%. EACeleb after SRE diarization performs best in all three cases and outperforms VoxCeleb by 36.78%. When using the model trained on EACeleb after VoxCeleb diarization, results are similar to the model of EACeleb after SRE diarization. EACeleb after SRE diarization performs best and all cases of EACeleb can outperform VoxCeleb testing data.

Also, the EER of model-matched dataset, which refers to testing data for the model trained on the corresponding training data, also shows that EACeleb after SRE diarization is the cleanest data among the three. EER after SRE diarization on matched model has 17.363% EER, while EACeleb has 22.669% and EACeleb after VoxCeleb has 21.174%. So from the analysis before, it can also show that SRE model can best filter the noisy part of EACeleb out and make the dataset afterwards cleanest among the three. Also, from the results compared with VoxCeleb testing data, our approach can effectively lower the EER using speaker diarization.

CHAPTER 4
CONCLUSION

In this thesis, we propose to use speaker diarization after the VoxCeleb data collection procedure to make the output cleaner. Doing this, we introduce the pipeline of our approach and collect a dataset named EACeleb with speakers in east Asian countries. In order to measure the cleanness of the collected dataset, we use automatic speaker verification to get the equal error rate result of different models and datasets. From our experiments, speaker diarization can effectively filter out parts that have noisy speakers, target speaker speaking in abnormal situations such as singing, target speaking with background music, etc. EER results show that EACeleb after speaker diarization can achieve 25.63% EER reduction compared with original EACeleb. And using speaker diarization, EACeleb can outperform the model baseline in certain cases. Also, from the model trained on EACeleb, all cases outperform VoxCeleb testing data and can achiheve up to 36.78% EER reduction. Our work aims at improving the cleanness of VoxCeleb data collection and do a complementary work to VoxCeleb by collecting a dataset for east Asian celebrities, thus adding more languages and characteristics to VoxCeleb.

Appendices

APPENDIX A

EXPERIMENTAL EQUIPMENT

In the data collection process, we used two NVIDIA GeForce GTX 1070 GPU with 8 GB memory. In the Kaldi processing stage, we used a personal computer with Intel Core i7-6800k CPU and 64 GB memory to do the speaker diarization and speaker verification, as well as our x-vector verification model training.

APPENDIX B

LIST OF NAMES OF SPEAKERS IN EACELEB

Following are names of the celebrity speakers in EACeleb in corresponding languages:

Chinese speakers:

杨洋	易中天	黄轩	刘震云	杨紫	闫妮
樊登	李开复	梅婷	丁彦雨航	郭碧婷	许嵩
容祖儿	范玮琪	陈乔恩	郭晶晶	陈数	美心
蒋雯丽	范志毅	洪昭光	马思纯	李静	陶虹
谭咏麟	华春莹	窦文涛	梁博	刘谦	林更新
薛蛮子	谭元元	海岩	唐骏	老狼	柳云龙
丁健	杜海涛	张亚勤	张继科	郑恺	戚薇
赵本山	杜鹃	李思思	杨超越	孙红雷	景甜
孙亚芳	李嘉欣	倪妮	冯小刚	徐小平	岳云鹏
佘诗曼	吴秀波	周华健	李云迪	刑菲	林书豪
金城武	杨丞琳	陈凯歌	耿爽	张惠妹	王宝强
张静初	任达华	蔡卓妍	张亮	刘翔	李铁
韩寒	钟欣桐	马琳	李宗盛	靳东	史玉柱
郭富城	姚晨	孙海英	英达	郭敬明	张天爱
王菲	邹廷威	张威	陈宝国	成龙	林志玲
黎明	陈好	言承旭	李幼斌	蔡徐坤	陈学冬
大张伟	陈道明	赵丽颖	姜文	罗晋	白宇
张怡宁	李锐	江疏影	黄豆豆	吴昕	井柏然
李娜	莫文蔚	林宥嘉	高圆圆	尼格买提	朱芳雨
李响	崔永元	康辉	胡军	郑洁	沈南鹏

郑嘉颖	蒋勤勤	孙继海	古天乐	杨澜	张艺谋
王力宏	陈晓	郭德纲	邹凯	吴宗宪	雷军
范冰冰	张钧甯	刘嘉玲	田亮	张朝阳	宋丹丹
韩红	张一山	裴蓓	毕福剑	郑爽	李斯羽
杨幂	谢娜	马苏	杨钰莹	赵又廷	彭昱畅
张连伟	孙楠	吕燕	梁文冲	李国庆	唐嫣
徐峥	崔健	徐熙娣	李小璐	华晨宇	沈梦辰
丁磊	吴敏霞	周涛	梁静茹	刘玉栋	黄磊
林俊杰	杨颖	张玉宁	金晨	陈赫	苏打绿
胡兵	张蕾	郑伊健	陈楚生	李双江	何超琼
白敬亭	迪丽热巴	李小冉	钟汉良	李明远	王朔
徐若	韩庚	谢楠	濮存昕	周鸿	李彦宏
宁静	刘若英	孙骁骁	李亚鹏	刘亦菲	毛不易
颜丹晨	邵佳一	李沁	张敬轩	窦唯	李小鹏
小沈阳	陆兆禧	刀郎	王一博	柳传志	朴树
许巍	刘烨	范伟	张牧野	那英	李一男
马龙	刘强东	吴奇隆	张子枫	阮经天	刘仪伟
彭于晏	罗志祥	张铁林	王治郅	马天宇	吴亦凡
李欣汝	林志颖	谢霆锋	何炅	桂纶镁	贝索斯
李冰冰	郑智	瞿颖	小s	赵薇	邓伦
苏有朋	甄子丹	鞠婧	汤唯	李咏	陈琳
娄艺潇	戴军	杨威	周冬雨	李湘	李艾
张学友	郝海东	钟欣潼	朱一龙	陈妍希	王哲林
郭晓冬	仲满	朱桢	郑元畅	姜培林	倪萍
陈一丹	郭采洁	刘德华	王晓晨	欧阳娜娜	罗永浩
李安	刘恺威	海清	王子文	张菲	冯远征
郎朗	宁浩	黄子韬	李少红	吴镇宇	王源

刘欢	彭蕾	李维嘉	王励勤	高希希	郑钧
大鹏	魏大勋	王楠	求伯君	吴京	王小川
王志东	方琼	任正非	蔡康永	池莉	陈建斌
张艺兴	陈鲁豫	宋佳	黄海波	张歆艺	胡杏儿
马未都	欧阳夏丹	霍建华	刘诗诗	张纪中	汪涵
胡歌	杨恭如	何润东	马化腾	周慧敏	黄圣依
李春江	方大同	武磊	于丹	陈嘉桦	徐新
李霞	陈华	张智霖	唐国强	李易峰	朱骏
黄健翔	孙杨	杨坤	张韶涵	冯绍峰	周杰伦
章泽天	王嘉尔	邓超	吉克隽逸	阿桑奇	张丰毅
杜淳	李庚希	黄奕	胡彦斌	庾澄庆	佟大为
王凯	董方卓	吴谨言	朱丹	刘雯	撒贝宁
董卿	Angelababy	雪村	杨千	秦岚	杜丽
吴宣仪	周星驰	周传雄	巩俐	田馥甄	王杰
常昊	宁丹琳	林峰	马伊	施密特	黄晓明
陈燮霞	斯琴高娃	黄景瑜	吕丽萍	陈奕迅	何洁
夏雨	张若昀	徐帆	柴静	茅侃侃	汪峰
鹿晗	林丹	方滨兴	徐铮	张杰	张靓颖
王俊凯	周迅	张雨绮	蒋欣	贾跃亭	
刘慈欣	张柏芝	王皓	古力娜扎	李永波	
徐静蕾	杨致远	柯蓝	刘涛	易烊千玺	
宋茜	谢杏芳	李晨	秦海璐	吴莫愁	

Japanese speakers:

中嶋朋子	新井浩文	姜暢雄	朝倉あき	瀬戸康史	山崎育三郎
佐津川愛美	河相我聞	矢田亜希子	生稲晃子	君島十和子	石原真理子
水野真紀	寺島しのぶ	牧瀬里穂	板谷由夏	堀北真希	峯田和伸

泉政行
山本涼介
榮倉奈々
天海祐希
佐倉しおり
吉沢悠
市川由衣
渡辺梓
原田知世
川村陽介
一色紗英
井ノ原快彦
永山絢斗
坂井真紀
ケイン・コスギ
内山理名
池松壮亮
大野いと
成宮寛貴
杉浦幸
松下由樹
松本穂香
森崎ウィン
秋本祐希
和久井映見
シャーロット・ケイ・フォックス
松田聖子
村川絵梨
東ちづる
佐藤隆太
広田レオナ
伊藤かずえ
小池里奈
松尾敏伸
宮本真希
内田有紀
小林涼子
小島藤子
森尾由美
佐々木希
山田まりや
渡辺満里奈
小泉孝太郎
仁科克基
半田健人
和田正人
吉野紗香
中山美穂
山村紅葉
中村由真
片瀬那奈
安田顕
岡田義徳
生田斗真
床嶋佳子
柴咲コウ
松田龍平
星野真里
加藤夏希
冨手麻妙
横山めぐみ
大野智
小田かおる
永作博美
尾上松也
小雪
三根梓
山下智久
土屋太鳳
松田翔太
本郷奏多
杏
ふせえり
中山エミリ
檜木直也
剛力彩芽
木内美穂
安達祐実
常盤貴子
原田龍二
大河内奈々子
浅香航大
瀬戸朝香
佐藤康恵
塚地武雅
石原さとみ
田丸麻紀
横山めぐみ
勝野雅奈恵
田畑智子
加藤ローサ
向井理
三浦春馬
中村倫也
本田なお
中村映里子
本仮屋ユイカ
志田未来
神楽坂恵
石橋杏奈
生田智子
小林聡美
櫻井翔
波岡一喜
深津絵里
マギー
松嶋菜々子
前川泰之
合田雅吏
神田沙也加
松たか子
丸高愛実
三浦涼介
北村有起哉
平山浩行
浅野ゆう子
桜井幸子
川島海荷
山本耕史
柄本時生
松井玲奈
井上和香
林遣都
加賀美セイラ
仙道敦子
新納慎也
仲野太賀
井上真央
加藤雅也
橋本甜歌
伊勢谷友介
窪塚俊介
大森南朋
真木よう子
優香
山下容莉枝
塚本高史
徳重聡
田村亮
山下リオ
杉田かおる
加藤晴彦
斉藤慶子
渡辺典子
今井美樹
仁科仁美
和希沙也
中川翔子
加藤貴子
乙葉
山崎裕太
堀内敬子
大塚寧々
小沢真珠
矢野浩二

安田美沙子
チ
柏原収史
正名僕蔵
伊藤歩
山本太郎
吉高由里子
中尾明慶
石田ひかり
三津谷葉子
熊谷真実
徳永えり
中江有里
波瑠
千堂あきほ
大後寿々花
片桐仁
安倍なつみ
石野真子
新田真剣佑
瀧内公美
桂木文
柏原崇
IZAM
森川葵
大泉洋
柳ゆり菜
山口馬木也
加納みゆき
市川実日子
岸本加世子
忽那汐里
城田優
毬谷友子
柏木由紀
柄本佑
成海璃子
小松菜奈
大場久美子
相楽樹
池内博之
佐野瑞樹
市川猿之助
小泉今日子
知英
入山法子
夏帆
井出卓也
（4代目）
島崎遥香
川岡大次郎
森下涼子
加藤紀子
有森也実
清水美沙
真野恵里菜
江角マキコ
勝地涼
森口彩乃
大西結花
小嶺麗奈
武田真治
仲間リサ
堺雅人
マイコ
森カンナ
三浦理恵子
窪田正孝
柳楽優弥
宮崎香蓮
ルビー・モ
水野美紀
水崎綾女
吉沢亮
吉岡里帆
富田靖子
レノ
田中美里
ムロツヨシ
栗山千明
今野浩喜
滝沢沙織
洞口依子
吉倉あおい
友近
井上芳雄
広末涼子
田中美奈子
小川菜摘
沢尻エリカ
清水富美加
木南晴夏
宍戸留美
尾野真千子
宅間孝行
山口紗弥加
工藤静香
福田沙紀
仲間由紀恵
二階堂ふみ
真飛聖
小川範子
杉原杏璃
渋谷飛鳥
濱田マリ
水原希子
松田美由紀
シルビア・
橋本環奈
児嶋一哉
伊藤健太郎
玉山鉄二
奥山佳恵
グラブ
木村多江
工藤阿須加
中村俊介
山本美月
岩佐真悠子
有沢比呂子
永井大
浜野謙太
坂口健太郎
上原多香子
松雪泰子
日南響子
窪塚洋介
加藤あい
広瀬アリス
ディーン・
木村了
千葉雄大
早見あかり
市川海老蔵
早見優
フジオカ
川村亜紀
宮澤寿梨
GACKT
（11代目）
初音映莉子
渡辺大
あめくみち
大政絢
佐藤江梨子
いしのよう
工藤夕貴
岡本杏理
こ
山本梓
武田久美子
こ
白石美帆
えなりかず
木村佳乃
永野芽郁
松山ケンイ
稲垣吾郎
吹石一恵
き
忍成修吾
新妻聖子

相武紗季	やべきょう	相田翔子	ユージ	向井亜紀	斉藤こず恵
堀江しのぶ	すけ	中島唱子	三船美佳	檀れい	中村優一
上遠野太洸	斎藤工	山田親太朗	松本莉緒	松尾諭	後藤久美子
安藤サクラ	大東駿介	夏菜	中谷美紀	尾上松緑	山本舞香
吉本多香美	岡本あずさ	稲森いずみ	田村淳	浅野忠信	古村比呂
有村架純	多部未華子	土屋アンナ	大和悠河	中村蒼	坂口憲二
岡田将生	森下悠里	中村勘九郎	市川実和子	山内圭哉	中村ゆりか
小澤征悦	河合美智子	（6代目）	沢口靖子	小栗旬	加瀬亮
反町隆史	喜多嶋舞	川栄李奈	中山忍	佐藤寛子	本田翼
三倉茉奈	仲里依紗	早織	水嶋ヒロ	マキタス	つみきみほ
武井咲	杉咲花	桐谷健太	大倉忠義	ポーツ	田中美保
優希美青	大和田美帆	壇蜜	吉本実憂	岡本玲	姿月あさと
中村ゆり	田中律子	石田ゆり子	上白石萌音	上野樹里	
山本裕典	早乙女友貴	はしのえみ	大野拓朗	本田理沙	

Korean speakers:

엘	민호	이찬	김빈	이보영	박초롱
주현	유퉁	바로	고수	최완정	기주봉
조춘	권율	이준	장용	박정학	정영숙
민욱	임호	장혁	이완	김우빈	최성원
류담	신국	이훈	장광	이정길	김이지
남경	임혁	주원	김진	이정신	조은숙
현빈	지성	성훈	유일	손은서	조연우
공유	윤박	현석	신구	이다희	조우리
재희	전운	원빈	거룡	방민아	최명길
지수	진영	랑연	이주현	유아인	이원근
준호	김권	고윤	정유진	오민석	조수향

신은수	박예진	임은경	최유화	문오장	정혜인
이도현	이초희	김지민	이인혜	김병세	박순천
최준용	조재현	맹세창	정인선	이민호	강남길
박정아	최동준	노희지	배종옥	태현실	박진주
서영화	박신양	이민우	박소은	오태경	이태임
손병호	남궁원	최화정	박세영	박병호	조경환
박신혜	김동규	윤용현	오경수	박세완	류시원
옥소리	김지현	이병준	김혜윤	이은주	류수영
김영란	이정재	박정철	김승우	장정희	나해령
전소민	한석규	박시후	선우선	민지아	오의식
박건형	김새론	박보검	류덕환	김정운	김성민
백옥담	양정아	김지미	김찬우	김흥수	고보결
신성우	이이경	이휘향	이선빈	이수나	김민진
서지훈	오연수	민지영	이동준	문정숙	박지영
최태준	박은빈	전국환	안성기	박찬환	박혁권
이대근	이용주	황정리	최정원	김태우	연규진
임대호	손태영	홍요섭	김정균	이민지	이유비
이준혁	윤주상	박보영	정우식	박규채	금새록
김영배	한가인	유인나	윤은혜	이지훈	이영진
신하균	김혜성	하나경	서은우	조윤우	고준희
이시영	옥자연	문보령	송새벽	김갑수	고세원
김무생	유세례	여진구	김상순	김예원	임성민
문채원	송창의	심지호	강성진	심은진	전혜빈
김수로	이수경	한그루	김유석	이현경	하승리
전지현	서하준	최성재	하석진	이제훈	이민기
장희진	이병욱	유혜리	손성윤	안재홍	이동욱
이준영	박철호	최무룡	박형식	최영완	진지희

홍기훈 박진성 김동현 이낙훈 안내상 이덕화
강기영 구혜선 박지윤 권유리 이상인 이혜미
채정안 박창민 유오성 이태곤 정일우 고소영
윤주희 정재용 김홍파 조덕현 최현호 송옥숙
류효영 이재은 임지은 김법래 박수영 전계현
김영인 송용태 채민서 문가영 서주현 김희갑
주지훈 심은하 홍경인 김명곤 김병옥 연제욱
박준규 권해성 지일주 장승조 이재용 주진모
강민경 전양자 이동휘 강부자 고현정 김기두
김민교 류화영 강문영 김형범 김병기 조승우
윤주만 김가희 정겨운 장진영 서태화 이정진
김지수 이희진 강수연 임원희 소유진 심은경
조복래 설경구 이나윤 길해연 이수혁 신혜선
하희라 안재욱 박윤배 이주희 이시아 김정태
김상호 홍지윤 소이현 김유리 나한일 이일화
김성균 김윤진 이상윤 정우성 김상중 정석원
정선경 민도희 박지예 한재석 김단율 전수경
이윤지 박하선 허준호 조미령 갈소원 전배수
강태성 이재원 정보석 고아성 정다빈 김재화
김소영 박인환 오승윤 남보라 서정연 노현희
강승원 권성덕 유해진 독고성 정준호 김가은
황보라 김영애 임시완 오현경 민찬기 하유미
표예진 김원희 김서라 김광현 고나은 김보미
박신우 송채환 유태웅 박영록 강은비 류승범
권해효 남경읍 심이영 이영유 김재중 최수종
서예지 김선영 봉태규 조민기 도은비 손호균
박수진 김을분 이기광 김준성 손예진 주호성

정호빈	김자옥	전인택	서범석	이영은	이혜영
오연아	이원종	김혜자	김영민	박지연	조문정
안재환	김현수	안소희	문근영	오세정	허영생
김희선	최진영	김정화	임수향	김준한	이예춘
최수영	조한선	김래원	이태환	박민우	조상구
이정헌	김민주	이장우	서영주	추수현	정애란
정채연	임윤아	이종혁	김민희	한진희	강다현
최여진	지윤호	이영애	이대엽	천우희	김창완
한지은	여욱환	신은경	안보현	이수빈	유승호
이시언	김환희	박윤재	권은수	박용하	고은미
인교진	김가연	구재이	이재윤	이범수	유호린
박선영	박탐희	문수빈	현승민	윤종훈	박혜진
최덕문	곽도원	김현중	김민준	임지연	허이재
최무성	서영희	조현재	문예원	전무송	조혜정
김선혁	송승환	진예솔	조형기	전도연	차수연
강인덕	손지현	강성연	신현준	유준상	안재현
문희경	조달환	김남길	윤문식	임슬옹	김인문
류태준	이종구	박민지	박해미	김규리	이재룡
배성우	이성경	이유리	강지환	판유걸	서준영
신동미	정링컨	정은채	김지원	연민지	남성진
김지훈	옥주현	윤지민	오대환	한지선	진기주
금보라	정명환	최민용	윤찬영	박원숙	이연수
김형준	경수진	연우진	서주성	남능미	이다해
한유이	김보성	데프콘	김시후	홍광호	손호준
이소연	고훈정	박은혜	이미연	윤상현	백승희
정수영	이요원	권상우	이열음	이혁재	정혜영
이태란	한지안	민영기	방은희	안병경	한인수

강경준	김주승	천호진	장신영	강리나	윤홍빈
황정음	황승언	최재성	마동석	정성영	이효정
이진아	오영실	박종관	이주승	이아현	서권순
박은석	박하나	박길라	한혜숙	김형자	정승호
유인촌	정욱진	윤영준	김지은	성동일	김승호
변영훈	김윤석	고두심	소지섭	김윤지	고경표
정준하	문천식	서민정	유혜정	김고은	박주미
남성훈	명로진	안선영	최윤소	류승룡	윤다훈
이규형	김수정	이승연	박상면	권혁수	권소현
김승수	박동빈	홍지민	김보라	유다인	송원석
채수빈	이수민	한선화	한보배	김강우	윤동환
심형탁	임현식	안세하	김성환	노주현	이윤미
박해일	김철기	박정화	한혜진	이유영	오인혜
강하늘	손나은	김예령	정은지	황찬성	정가은
김지성	김석훈	홍수아	문성근	박용우	배도환
김학선	김아중	양동근	박영지	김규종	김남진
성유리	전여빈	전노민	이영아	고주원	윤기원
천이슬	이혜리	도상우	정예진	이응경	송일국
이나영	서현철	최은희	윤철형	박진희	정경호
김정난	심혜진	신은정	이성재	오미연	김유정
송영창	정혜성	설인아	이태성	진서연	김남주
오아연	박병선	최시원	정은표	박광현	허정은
무진성	공형진	박강현	이민정	김다솜	임성언
이상우	홍수현	이미영	강예원	윤보라	최윤영
윤유선	김정현	홍아름	김현주	서지영	남우현
김민정	서민지	김용림	송삼동	정애리	이예림
강소라	김동희	배다빈	권오중	장소연	유하나

이희준	박성훈	김슬기	유동근	남궁민	황동주
안소영	김진엽	유민규	이광수	이현우	김선동
홍태의	박용진	오재무	최대철	한혜린	이규한
백성현	서범식	노정의	최성국	홍학표	최재호
오윤아	양희경	박원상	한보름	민효린	연정훈
최유라	김준수	강두리	이재황	선동혁	윤손하
이창환	서인국	박민하	임영규	김동주	손여은
차예련	이신애	허정민	박광정	최불암	채시라
임주은	염정아	황선희	이문식	최주봉	윤소이
윤계상	한상진	최일화	노영학	유호정	정시아
임동진	최필립	채상우	윤현민	안문숙	최우식
한서진	백일섭	차인표	강예솔	함은정	오지명
강성아	유인영	배수지	명계남	추자현	이하늬
이자인	이정현	정수정	박희순	원미경	박준금
김태연	최강희	오수미	진세연	윤희석	김일우
예지원	이영훈	임강성	이병헌	장동윤	조재윤
송재호	최재원	김소은	신신애	정태우	손창호
이연희	심희섭	장희령	박지현	최지호	한수연
길용우	김승대	백진희	남규리	이다인	유연미
송재림	이지영	최종원	최정우	오정세	이재포
장동건	오지은	정호근	민영원	이은형	지이수
박성웅	문회원	이미숙	이시원	유지태	강태오
김뢰하	서은광	기은세	이주연	조민아	윤세웅
이상숙	최재환	강동원	박해진	유연석	송지효
도경수	홍진희	나영희	전성우	김희애	박시은
김애경	전미선	공승연	이민영	송선미	김나운
최철호	이채영	이다윗	황정순	최은주	고아라

이준기	서현진	송혜교	김민경	최자혜	독고영재
김대명	손현주	김도연	김성령	예수정	노수산나
김지우	신현빈	신재하	성지루	오달수	황우슬혜
황은정	김희정	박재정	변요한	최동균	선우용여
정려원	류혜영	육동일	김다현	서도영	
이필모	한소희	장성원	이광기	전혜원	
강지섭	오승은	이상희	김승훈	조희봉	
김무열	권현상	노영국	김정훈	최다니엘	
권화운	서강준	정동남	신정근	장미인애	

REFERENCES

[1] D. A. Reynolds, "Speaker Identification and Verification Using Gaussian Mixture Speaker Models," *Speech Communication*, vol. 17, no. 1-2, pp. 91–108, 1995.

[2] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker Verification Using Adapted Gaussian Mixture Models," *Digital Signal Processing*, vol. 10, no. 1-3, pp. 19–41, 2000.

[3] M. McLaren, L. Ferrer, D. Castan, and A. Lawson, "The Speakers In the Wild (SITW) Speaker Recognition Database.," in *Interspeech*, 2016, pp. 818–822.

[4] S. O. Sadjadi, T. Kheyrkhah, A. Tong, C. S. Greenberg, D. A. Reynolds, E. Singer, L. P. Mason, and J. Hernandez-Cordero, "The 2016 NIST Speaker Recognition Evaluation.," in *Interspeech*, 2017, pp. 1353–1357.

[5] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: A Large-scale Speaker Identification Dataset," *arXiv preprint arXiv:1706.08612*, 2017.

[6] J. H. Hansen and T. Hasan, "Speaker Recognition by Machines and Humans: A Tutorial Review," *IEEE Signal Processing Magazine*, vol. 32, no. 6, pp. 74–99, 2015.

[7] M. R. Hasan, M. Jamil, M. Rahman, *et al.*, "Speaker Identification Using Mel Frequency Cepstral Coefficients," *Variations*, vol. 1, no. 4, 2004.

[8] H. Hermansky, "Perceptual Linear Predictive (PLP) Analysis of Speech," *Journal of the Acoustical Society of America*, vol. 87, no. 4, pp. 1738–1752, 1990.

[9] M. Todisco, H. Delgado, and N. Evans, "Constant Q Cepstral Coefficients: A Spoofing Countermeasure for Automatic Speaker Verification," *Computer Speech & Language*, vol. 45, pp. 516–535, 2017.

[10] K Li and E Wrench, "An Approach to Text-Independent Speaker Recognition with Short Utterances," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 8, 1983, pp. 555–558.

[11] D. A. Reynolds and R. C. Rose, "Robust Text-Independent Speaker Identification Using Gaussian Mixture Speaker Models," *IEEE Transactions on Speech and Audio Processing*, vol. 3, no. 1, pp. 72–83, 1995.

[12] J.-L. Gauvain and C.-H. Lee, "Maximum A Posteriori Estimation for Multivariate Gaussian Mixture Observations of Markov Chains," *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 2, pp. 291–298, 1994.

[13] W. M. Campbell, D. E. Sturim, and D. A. Reynolds, "Support Vector Machines Using GMM Supervectors for Speaker Verification," *IEEE Signal Processing Letters*, vol. 13, no. 5, pp. 308–311, 2006.

[14] R. Kuhn, J.-C. Junqua, P. Nguyen, and N. Niedzielski, "Rapid Speaker Adaptation in Eigenvoice Space," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 6, pp. 695–707, 2000.

[15] P Kenny, M Mihoubi, and P Dumouchel, "New MAP Estimates for Speaker Recognition," in *Proc. of EUROSPEECH*, 2003.

[16] P. Kenny, "Joint Factor Analysis of Speaker and Session Variability: Theory and Algorithms," *CRIM, CRIM-06/08-13*, vol. 14, pp. 28–29, 2005.

[17] N. Dehak, P. Kenny, R. Dehak, O. Glembek, P. Dumouchel, L. Burget, V. Hubeika, and F. Castaldo, "Support Vector Machines and Joint Factor Analysis for Speaker Verification," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2009, pp. 4237–4240.

[18] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez, "Deep Neural Networks for Small Footprint Text-dependent Speaker Verification," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 4052–4056.

[19] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust DNN Embeddings for Speaker Recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 5329–5333.

[20] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-End Factor Analysis for Speaker Verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2010.

[21] S. J. Prince and J. H. Elder, "Probabilistic Linear Discriminant Analysis for Inferences About Identity," in *IEEE 11th International Conference on Computer Vision*, 2007, pp. 1–8.

[22] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep Speaker Recognition," *arXiv preprint arXiv:1806.05622*, 2018.

[23] S. E. Tranter and D. A. Reynolds, "An Overview of Automatic Speaker Diarization Systems," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 5, pp. 1557–1565, 2006.

[24] X. Anguera, S. Bozonnet, N. Evans, C. Fredouille, G. Friedland, and O. Vinyals, "Speaker Diarization: A Review of Recent Research," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 2, pp. 356–370, 2012.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[26] J. S. Chung and A. Zisserman, "Out of Time: Automated Lip Sync in the Wild," in *Asian Conference on Computer Vision*, Springer, 2016, pp. 251–263.

[27] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, "Deep Neural Network Embeddings for Text-Independent Speaker Verification.," in *Interspeech*, 2017, pp. 999–1003.

[28] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, *et al.*, "The Kaldi Speech Recognition Toolkit," in *IEEE Workshop on Automatic Speech Recognition and Understanding*, 2011.